

---

# **frequir**

***Release 0.1***

**Jesse Wood**

**Mar 04, 2022**



**CONTENTS:**

<b>1</b>	<b>Home</b>	<b>3</b>
1.1	Time domain . . . . .	3
1.2	Installation . . . . .	4
1.3	Testing . . . . .	4
1.4	References . . . . .	4
<b>2</b>	<b>freqrir</b>	<b>5</b>
2.1	freqrir module . . . . .	5
2.2	helper module . . . . .	6
2.3	timerir module . . . . .	8
<b>3</b>	<b>License</b>	<b>11</b>
<b>4</b>	<b>Contact</b>	<b>13</b>
<b>5</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>
	<b>Index</b>	<b>19</b>



Generate a room impulse response in the frequency domain.



## 1.1 Time domain

Plot a typical impulse response for a room 80 x 12 x 100 sample lengths long. Wall reflection coefficients were all 0.9, ceiling and floor coefficients were 0.7. Source and receiver were at (30, 100, 40) and (50, 10, 60) sample periods [1].

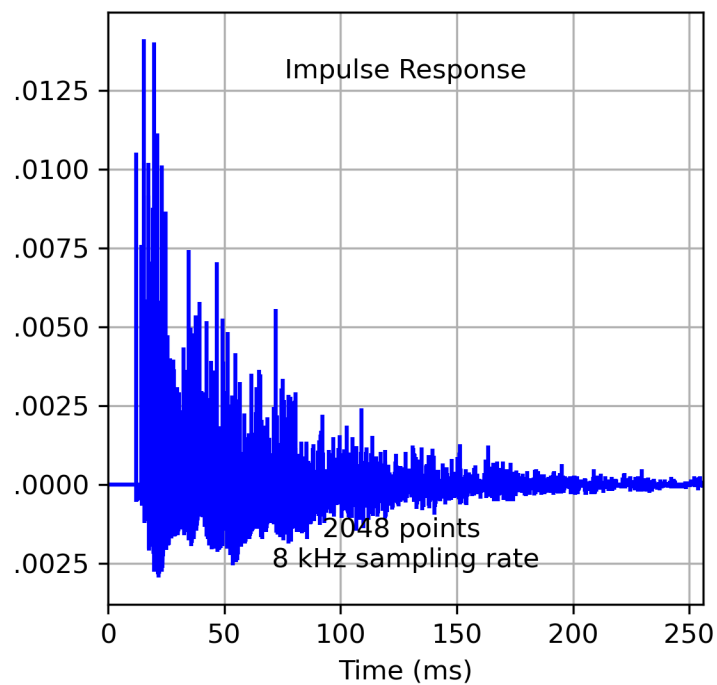


Fig. 1: Room impulse response in time-domain

## 1.2 Installation

The python libraries necessary to run this can be installed using `pip` as follows

```
$ pip install .  
$ pip install -r requirements.txt
```

## 1.3 Testing

The unit tests are located in the `tests` directory, they can be run from the root directory

```
$ python -m unittest discover -s tests
```

## 1.4 References

1. Allen, J. B., & Berkley, D. A. (1979). Image method for efficiently simulating small-room acoustics. The Journal of the Acoustical Society of America, 65(4), 943-950. [Available](#)
2. Lehmann, Eric A., and Anders M. Johansson. "Prediction of energy decay in room impulse responses simulated with an image-source model." The Journal of the Acoustical Society of America 124.1 (2008): 269-277. [Available](#)



## FREQRIR

## 2.1 freqrir module

`freqrir.freqrir.frequency_rir(receivers, source, room_dimensions, betas, points, sample_frequency, frequency, c=304.8, T=0.0001, order=-1)`

Calculate room impulse response in the frequency domain.

### Parameters

- **receiver** (*list[float] with shape (3,)*) – Receiver location in sample periods (s).
- **source** (*list[float] with shape (3,)*) – Source location in sample periods (s).
- **room\_dimensions** (*list[float] with shape (3,)*) – Room dimensions in sample periods (s).
- **betas** (*float np-array with shape (3,2)*) – Absorption coefficients. Walls: left, right, front, back, floor, ceiling.
- **points** (*int*) – Number of points, which determines precisions of bins.
- **sample\_frequency** (*float*) – Sampling frequency or sampling rate (Hz).
- **frequency** (*float*) – Frequency of interest (Hz).
- **c** (*float, optional*) – Speed of sound (m/s). Defaults to 304.8 m/s (i.e. 1 ft/ms) (Allen 1979).
- **T** (*float, optional*) – Sampling period (s). Defaults to 1E-4 s (i.e. 0.1 ms) (Allen 1979).
- **order** (*int, optional*) – Maximum order of reflections. Defaults to -1 (i.e. all reflections).

**Returns** A pressure wave in the frequency domain.

**Return type** pressure (complex)

**Raises** **ValueError** – If source and receiver are too close together (i.e. within 0.5 sampling periods).

## 2.2 helper module

`freqrir.helper.distance_for_permutations(receiver, source, room_dimensions, vector_triplet)`

Computes the distances between the reciever and the eight image source permutations.

**Parameters**

- **receiver** (*list[float]*) – Reciever position.
- **source** (*list[float]*) – Source position.
- **room\_dimensions** (*list[float]*) – Room dimensions.
- **vector\_triplet** (*list[float]*) – Vector triplet (n,l,m) (Allen 1979).

**Returns** The distances between the reciever and the eight image source permutations.

**Return type** distances (list[float] with shape (8,))

**Examples**

```
>>> distance_for_permutations(np.array([0,0,0]), np.array([1,1,1]), np.array([5,5,5]), np.array([0,0,0]))[0]
1.7320508075688772 # Take the first element of the list.
```

`freqrir.helper.distance_from_offset(r, offset=[0, 0, 0])`

Compute the distances for the reciever locations from the offset.

This method generates a density plot for the distances from the offset. The purpose of this method is to verify that the distances are being generated with a uniformly distributed magnitude from the offset (i.e. the center of the point cloud).

**Parameters**

- **r** (*Array-like*) – Array of reciever locations.
- **offset** (*list[float]*) – Offset from origin for center of point cloud. Default is [0, 0, 0] (origin).

**Returns** Array of distances.

**Return type** d (Array-like)

`freqrir.helper.meters_to_sample_periods(x, sample_rate, c=304.8)`

Convert a measurement from meters to sample periods.

**Parameters**

- **x** (*float*) – A measurement in meters (m).
- **c** (*float*) – Speed of sound (m/s). Defaults to 304.8 m/s (i.e. 1 ft/ms).
- **T** (*float*) – Sampling period (s). Defaults to 1E-4 s (i.e. 0.1 ms).

**Returns** A measurement in sample periods (s).

**Return type** x (float)

## Examples

```
>>> meters_to_sample_periods(3.048, 8000) # 1.2 meters (m)
80 # sample periods (s)
```

`freqrir.helper.plot_frequency_rir(rir, points, frequency, save=None)`

Plot room impulse response in the frequency domain.

### Parameters

- **rir** (*list[complex]*) – A pressure wave in the frequency domain.
- **points** (*int*) – The number of points.
- **frequency** (*int*) – Sampling rate (Hz)
- **save** (*str, optional*) – Path to save file to. Defaults to None.

`freqrir.helper.plot_recievers(r, projection='2d')`

Plot the reciever locations.

### Parameters

- **r** (*Array-like*) – Array of reciever locations.
- **projection** (*str*) – Projection of the reciever locations. Default is 2d.

`freqrir.helper.plot_time_rir(rir, points, f, rt60, save=None)`

Plot room impulse response in the time domain.

### Parameters

- **rir** (*list[complex]*) – A pressure wave in the frequency domain.
- **points** (*int*) – The number of points.
- **rt60** (*float*) – The reverberation time (RT60) of the room.
- **f** (*int*) – Sampling rate (Hz)
- **save** (*str, optional*) – Save the plot to a file.

`freqrir.helper.sample_period_to_feet(x, sample_frequency, c=1000)`

Convert a measurement from sample periods to meters.

### Parameters

- **x** (*float*) – A measurement in sample periods.
- **sample\_frequency** (*int*) – Sampling rate (Hz).
- **c** (*float, optional*) – Speed of sound (ft/s). Defaults to 1000 ft/ms (i.e. 304.8 m/s SI).

**Returns** A measurement in meters (feet).

**Return type** x (float)

### Examples

```
>>> sample_period_to_feet(80,8000) # 80 sample periods (s)
10 # ft
```

`freqrir.helper.sample_period_to_meters(x, sample_rate, c=304.8)`

Convert a measurement from sample periods to meters.

#### Parameters

- **x** (*float*) – A measurement in sample periods.
- **sample\_rate** (*int*) – Sampling rate (Hz).
- **c** (*float*) – Speed of sound (m/s). Defaults to 304.8 m/s (i.e. 1 ft/ms).

**Returns** A measurement in meters (m).

**Return type** x (float)

### Examples

```
>>> sample_period_to_meters(80, 8000) # 80 sample periods (s) at 8 kHz
3.048 # meters (m)
```

`freqrir.helper.sample_random_receiver_locations(n, radius, offset=[0, 0, 0])`

Sample a random receiver location from within a spherical point cloud.

#### Parameters

- **n** (*int*) – number of receiver locations to sample.
- **radius** (*float*) – radius of the point cloud.
- **offset** (*list[float]*, *optional*) – offset from origin for center of point cloud. Default is [0, 0, 0] (origin).

**Returns** Array of receiver locations.

**Return type** r (Array-like)

## 2.3 timerir module

`freqrir.timerir.high_pass_filter(pressures, points, sample_frequency)`

High-pass digital filter to wierd behaviour at low frequencies (i.e. 100 Hz).

#### Parameters

- **pressures** (*list[complex]*) – Pressure wave in the time domain.
- **points** (*int*) – The number of points.
- **sample\_frequency** (*float*) – Sampling frequency or sampling rate (Hz).

**Returns** Pressure wave with frequencies below cutoff removed.

**Return type** pressures (list[complex])

`freqrir.timerir.time_rir(receivers, source, room_dimensions, betas, points, sample_frequency, order=-1, c=304.8)`

Calculate room impulse response in the time domain.

#### Parameters

- **receivers** (*list[list[float]] with shape (N,3)*) – Reciever location(s) in sample periods (s).
- **source** (*list[float] with shape(3,)*) – Source location in sample periods (s).
- **room\_dimensions** (*list[float] with shape (3,)*) – Room dimensions in sample periods (s).
- **betas** (*float np-array with shape (3,2)*) – Absorbtion coefficients. Walls: left, right, front, back, floor, ceiling.
- **points** (*int*) – Number of points, which determines precisions of bins.
- **sample\_frequency** (*float*) – Sampling frequency or sampling rate (Hz).
- **c** (*float, optional*) – Speed of sound (m/s). Defaults to 304.8 m/s (i.e. 1 ft/ms) (Allen 1979).

**Returns** A pressure wave in the time domain.

**Return type** pressures (list[complex])

**Raises ValueError** – If source and receiver are too close together (i.e. within 0.5 sampling periods).

`freqrir.timerir.time_rir_slow(receiver, source, room_dimensions, betas, points, sample_frequency, c=304.8)`

Calculate room impulse response in the time domain.

#### Parameters

- **receiver** (*list[float] with shape (3,)*) – Reciever location in sample periods (s).
- **source** (*list[float] with shape(3,)*) – Source location in sample periods (s).
- **room\_dimensions** (*list[float] with shape (3,)*) – Room dimensions in sample periods (s).
- **betas** (*float np-array with shape (3,2)*) – Absorbtion coefficients. Walls: left, right, front, back, floor, ceiling.
- **points** (*int*) – Number of points, which determines precisions of bins.
- **sample\_frequency** (*float*) – Sampling frequency or sampling rate (Hz).
- **c** (*float, optional*) – Speed of sound (m/s). Defaults to 304.8 m/s (i.e. 1 ft/ms) (Allen 1979).

**Returns** A pressure wave in the time domain.

**Return type** pressures (list[complex])

**Raises ValueError** – If source and receiver are too close together (i.e. within 0.5 sampling periods).



**LICENSE**

MIT License

Copyright (c) 2022 Jesse Wood

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.





**CONTACT**

Questions? Please contact [j.r.h.wood98@gmail.com](mailto:j.r.h.wood98@gmail.com)



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### f

`freqrir.freqrir`, 5  
`freqrir.helper`, 6  
`freqrir.timerir`, 8



## INDEX

### D

`distance_for_permutations()` (in module *freqrir.helper*), 6  
`distance_from_offset()` (in module *freqrir.helper*), 6

### F

`freqrir.freqrir`  
    module, 5  
`freqrir.helper`  
    module, 6  
`freqrir.timerir`  
    module, 8  
`frequency_rir()` (in module *freqrir.freqrir*), 5

### H

`high_pass_filter()` (in module *freqrir.timerir*), 8

### M

`meters_to_sample_periods()` (in module *freqrir.helper*), 6  
module  
    `freqrir.freqrir`, 5  
    `freqrir.helper`, 6  
    `freqrir.timerir`, 8

### P

`plot_frequency_rir()` (in module *freqrir.helper*), 7  
`plot_recievers()` (in module *freqrir.helper*), 7  
`plot_time_rir()` (in module *freqrir.helper*), 7

### S

`sample_period_to_feet()` (in module *freqrir.helper*),  
    7  
`sample_period_to_meters()` (in module *freqrir.helper*), 8  
`sample_random_receiver_locations()` (in module  
    *freqrir.helper*), 8

### T

`time_rir()` (in module *freqrir.timerir*), 8  
`time_rir_slow()` (in module *freqrir.timerir*), 9